REMARKS

Present Status of Patent Application

This is a response to the non-final Office Action (Paper No. 3) of January 4, 2002. Upon entry of this response, claims 17-22 are added, and claims 1-22 are pending in the application. The Applicants respectfully request that the amendment filed herewith be entered and that there be reconsideration of the claims as amended.

The Office Action rejected claims 1-16 under 35 U.S.C. §102(a) as being anticipated by U.S. Patent No. 5,828,886, to Hayashi. For the reasons set forth herein, Applicants respectfully request reconsideration and withdrawal of these rejections.

Summary of Present Invention

To achieve the advantages and novel features, the present invention is generally directed to a system and method for efficiently passing compiler information at run time to hardware or software in an efficient way. The present invention is particularly useful for efficiently passing compiler information during code optimization or translation utilizing free or unused operand fields of instructions such as NOP, and encoding the compile time information in the unused operand field. This technique removes the time overhead for analyzing binaries or low-level programs and does not increase program code size. NOP instructions perform no operation and are generally used as filler or instruction place-holders. For example, NOP operations have an immediate operand that is not used.

The present invention provides a system and method for passing compile time information between a compiler and real-time operation of post-compile-time software. Briefly described, in architecture, the system can be implemented as follows. The preferred system of the present invention utilizes an unused NOP operand (a register usage bit vector) that is a vehicle (or communication channel) between a static compiler and a dynamic optimizer. Each bit in the vector represents a particular register and is used to indicate if the register may be live. The register usage bit vector in the unused NOP operand is used to make finding free registers easier during optimization.

The present invention can also be viewed as providing a method for passing compile time information between a compiler and real-time operation of post-compile-time software. In this regard, the method can be broadly summarized by the following steps: the compiler produces bit vectors for each basic block, (*i.e.*, subroutine, function, and/or procedure) and places the bit vector in the unused portion of the NOP instruction encoding. A bit in the vector represents a particular register. A bit is set if the register may be live at the location of the NOP instruction and allows the dynamic optimizer to determine if further analysis of the low-level code to determine whether the register is truly live is required. On the other hand, a zero (*i.e.*, unset) bit in the bit vector signals that the compiler does not use the corresponding register (*i.e.*, is a dead register) at the location of the NOP instruction, and therefore the register can be used by the dynamic optimizer.

Because the compiler stores the dead register information in the unused operand area, the analysis information can be accessed without making the low-level code larger. Should the low-level code not have a NOP instruction to encode register utilization information, the dynamic optimizer or other software can examine the operands of NOPs instructions in the surrounding basic blocks to deduce the missing information. A basic block is a collection of a sequence of instructions that are entered at the top of the sequence and exited at the bottom of the sequence.

An advantage of deducing this missing information is that the information is particularly useful in improving performance of dynamic optimizations performed at runtime.

This is because the analysis overhead directly reduces performance when performed. In the preferred method of the present invention, because the dynamic optimizations may inspect the unused NOP operands very quickly, the overhead is dramatically reduced to improve runtime performance.

In another embodiment, the compiler may pass hints to profiler software of what kind of feedback information is desired through the use of the unused NOP operands. Because of many other possible uses of this communication channel, the compiler has to annotate the low-level code binaries it produces to indicate what information is contained in the unused NOP operands.

Discussion of Rejections

The Office Action rejected claims 1-16 under 35 U.S.C. §102(a) as being anticipated by U.S. Patent No. 5,828,886, to *Hayashi*. For the reasons set forth below, Applicants respectfully traverse this rejection.

Independent Claims 1, 6, and 11

Independent claim 1 recites:

1. A register usage indicator system for efficiently signaling register usage in a computer program comprising a plurality of blocks of code, said register usage indicator system comprising:
   a code usage register ***contained within a NOP instruction*** in one of the plurality of blocks of code in the computer program, said code usage register comprising a plurality of storage bits; and
   a code register usage annotator for determining if each one of the plurality of registers is live in one of the plurality of blocks of code containing said NOP instruction.

(Emphasis added.)

Independent claim 6 recites:

6. A method to efficiently signal register usage in a computer program comprising a plurality of blocks of code, the method comprising the steps of:
determining which of a plurality of registers are live in one of the plurality of blocks of code in the computer program;
finding at least one NOP instruction in one of the plurality of blocks of code;
creating a code usage register having a plurality of storage bits ***in said at least one NOP instruction*** in one of the plurality of blocks of code; and
setting one of said plurality of storage bits for each one of the plurality of registers live in one of the plurality of blocks of code containing said NOP instruction.

(Emphasis added.)

Independent claim 11 recites:

11. A register usage indicator system for efficiently signaling register usage in a computer program comprising a plurality of blocks of code, said register usage indicator system comprising:
means for determining which of the plurality of registers are live in one of the plurality of blocks of code in the computer program;
means for finding at least one NOP instruction in said one of the plurality of blocks of code;
means for creating a code usage register ***in said at least one NOP instruction*** in said one of the plurality of blocks of code; and
means for setting one of a plurality of storage bits in said code usage register for each one of the plurality of registers live in said one of the plurality of blocks of code containing said NOP instruction.

(Emphasis added.)

Applicants respectfully submit that independent claims 1, 6, and 11 patently define over the cited reference for at least the reason that the cited reference fails to disclose or otherwise teach the features emphasized above.

The Office Action alleges that *Hayashi* anticipates the register indicator system of claims 1, 6, and 11. However, Applicants respectfully submit that *Hayashi* neither teaches nor discloses each and every element of the claimed invention, and, hence, *Hayashi* does not anticipate Applicants invention. It is well settled that "[a]nticipation requires the disclosure in a single prior art reference of each element of the claim under consideration." *W. L. Gore & Associates, Inc. v. Garlock Inc.*, 721 F.2d 1540 (Fed. Cir. 1983).

Here, Applicant respectfully asserts that *Hayashi* does not teach that a code usage register is ***contained within a NOP instruction***. While *Hayashi* does disclose a register allotting process and instruction scheduling process, Applicant contends that *Hayashi* ***does not use any portion of the NOP instruction*** to carry register information. As evidenced by every example and the entirety of *Hayashi*'s disclosure, the system of *Hayashi*'s never uses the NOP instruction to hold a code usage register. For example, the table in column 15, line 57 through column 16, line 24 shows twelve NOP instructions. However, Applicant respectfully asserts that ***none*** of the twelve NOP instructions are associated with any register information. Similarly, the table in column 18 line 46 through column 19, line 8 shows four NOP instructions, which are all devoid of register information. In similar fashion, Applicant respectfully declares that every NOP instruction in every one of *Hayashi*'s examples (*i.e.*, the tables in column 21, line 51 through column 22, line 25; column 22, line 35 through column 23, line 15; column 23, lines 25-63; column 24, lines 17-49; and column 24, line 65 through column 25, line 37) is devoid of register information.

Additionally, the entirety of the written description mentions the NOP instruction once, and only in the context of the NOP instruction being a demarcation point for the end of a series of instructions. Thus, the NOP instruction of *Hayashi* functions in the same manner as the typical NOP instruction of the prior art - as a "filler" or a "place-holder." Applicants respectfully submit that *Hayashi*, therefore, does not teach "a code usage register ***contained within a NOP instruction***" as claimed.

In contrast to *Hayashi*'s invention, claims 1, 6, and 11 of the pending application provide for a code usage register ***contained within a NOP instruction***. As described in

Applicants' disclosure (*e.g.*, page 3, line 16 through page 4, line 11; page 8, line 19 through page 9, line 8), the NOP of the present invention is not merely used as a "filler" or a "place-holder." Thus, Applicants respectfully submit that claims 1, 6, and 11 embody an invention that is patentably distinct from the invention of *Hayashi*.

Dependent Claims 2-5, 7-10, and 12-16

In as much as claims 2-5 depend, either directly or indirectly, from allowable independent claim 1 and incorporate all of the limitations of claim 1, Applicants respectfully submit that claims 2-5 are in condition for allowance. Also, since claims 7-10 depend from allowable claim 6 and incorporate all of the limitations of claim 6, Applicants respectfully submit that claims 7-10 are in condition for allowance. Furthermore, since claims 12-16 depend from allowable claim 11, Applicants respectfully submit that claims 12-16 are in condition for allowance. Hence, Applicants submit that, for at least this reason, dependent claims 2-5, 7-10, and 12-16 are patentable over the cited references.

New Claims 17-22

Applicants submit that new claims 17 through 22 are allowable as having the limitation that a code usage register is contained within a NOP instruction. New claims 17 through 22 are directed to a computer readable medium, and are companion claims corresponding in scope to the originally submitted system claims.

Prior Art Made of Record

The prior art made of record has been considered, but is not believed to affect the patentability of the presently pending claims.

CONCLUSION

Applicants respectfully submit that all claims are now in proper condition for allowance, and respectfully request that the Examiner advance this case to issuance. If, in the opinion of the Examiner, a telephonic conference would expedite the examination of this matter, the Examiner is invited to call the undersigned attorney at (770) 933-9500.

Respectfully submitted,

[illegible]

Robert E. Stachler

Registration No. 36,934

THOMAS, KAYDEN, HORSTEMEYER & RISLEY, L.L.P.
Suite 1750
100 Galleria Parkway N.W.
Atlanta, Georgia 30339
(770) 933-9500
(770) 951-0933 (Facsimile)
50814-1770 - RES